Corel® Paradox® 8

April Newsletter April 1, 1998

The Visual Query Builder (VQB)

This month's newsletter describes the new Visual Query Builder(VQB) in Corel®Paradox®. Developed at Borland as part of the Borland Database Engine, the VQB is state of the art in SQL query designers. The following topics will be explored:

- **The Visual Query Builder (VQB)**
- **Defining the Query's Data Model**
- **Specifying Query Criteria**
- **Selecting Fields to Display**
- **Grouping**
- **Sorting**
- **Query Properties**
- **Viewing, Modifying, Saving and Loading the SQL**
- **Subqueries**

This information about the VQB was prepared by Dan Ehrmann, one of the foremost experts in Paradox. It was first presented at Borland's 1997 Developer Conference.

Dan Ehrmann is the founder and President of Kallista. For more information about Kallista, please visit www.kallista.com.

The Visual Query Builder (VQB)

The Visual Query Builder (VQB) provides a complete and modern graphical surface that generates standard SQL. It is easy to learn for basic queries, and offers many options for advanced users.

The Visual Query Builder was designed with the following objectives in mind:

Easy to learn – the functionality is intuitive and you won't find any "dead-end" screens.
Easy to use – there are fewer keystrokes or mouse clicks required to do basic things.
Graphical – we use drag and drop, a data model diagram, Windows controls and a modern user interface.
Graduated complexity – beginners will benefit from the clean, simple user interface and advanced users will find more capabilities.
Powerful – you'll find support for the full range of query options in the SQL standard.
Compatible – no change in query by example (QBE) means compatibility with the current QBE.
Two-way tool – the addition of this tool means that a change in the graphical UI is reflected immediately in the SQL.
Extensible – this capability will allow for future enhancements.

Accessing the Tool

In Corel® Paradox® 8, the Visual Query Builder (VQB) is not positioned as the primary query interface. The old QBE is still there, and will be accessed when you create a new query. Even if you create a new SQL statement, you still get a basic text editor to begin with.

The VQB can only be invoked from the SQL Editor. To display the VQB, open a new SQL query, select **SQL | Send to Query Builder** from the menu.

You can also click on the equivalent toolbar icon (the arrow pointing to "SQL"). If you are creating a new SQL statement, you will see the following window:

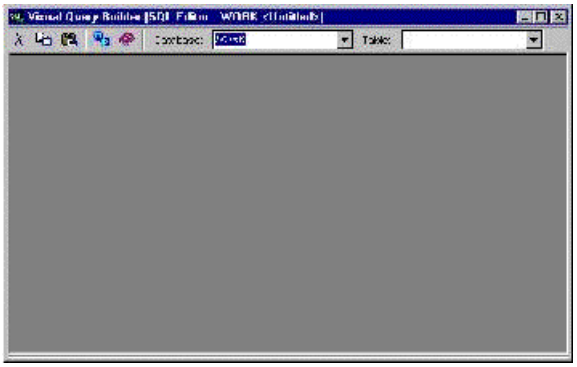


Figure 1: The Visual Query Builder before you start defining the query.

If you are editing an existing SQL statement, you will see the VQB with your SQL statement parsed and already loaded, as in figure 2.

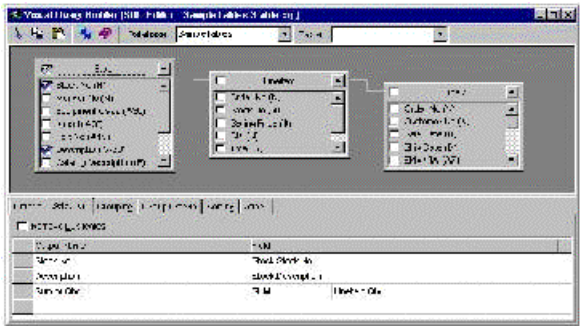
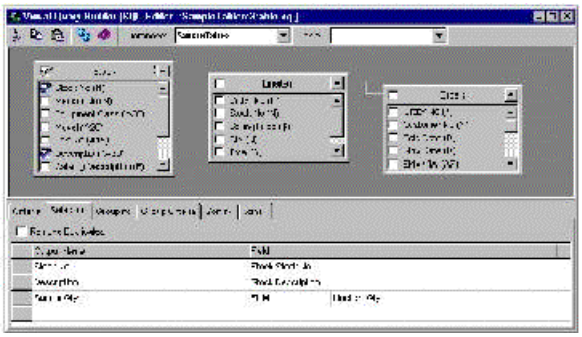


Figure 2: The Visual Query Builder with a query loaded.

While you are defining and editing the query, the SQL Editor window remains open in the background and is updated in real time! This is a true Two-Way Tool. To close the VQB, simply click its Close titlebar icon to return to the SQL Editor. From there, you can save or cancel the SQL statement, or specify SQL properties.

User Interface Overview

Notice the following elements in Figure 2.



- The top panel is the Data Model window. It allows you to specify the tables involved in the query, including multiple instances of the same table for a self-join. Join-lines express the links between tables. You can also select fields for inclusion in the query by clicking the check box on the left side of each field. (Specify all fields in the table by clicking the check box beside the table name.) Table aliases can be specified and tables can be shrunk to an icon or expanded to the full view shown here.
- The bottom panel is a series of notebook tabs reflecting the different elements of the SQL statement. Click on the appropriate tab to make it current.
- The **Criteria** tab allows you to express complex expressions for the WHERE clause of your query. It supports prompted expressions, SQL expressions entered directly, and EXISTS clauses that lead to subqueries. You can also specify multiple expressions, complex nesting and grouping relationships.
- The **Selection** tab allows you to specify the fields to appear in the result table. This tab supports fields and summaries and allows you to specify field aliases (AS clause). It

also allows you to control the physical order of fields in the result table.
- The **Grouping** tab defines group fields when you use aggregate expressions in the Selection tab. It is equivalent to the GROUP BY clause of your query. The contents of this tab are usually specified automatically by the VQB as you define fields in the Selection tab.
- The **Group Criteria** tab defines selection criteria on aggregate results. It is equivalent to the HAVING clause in SQL.
- The **Sorting** tab allows you to define the overall sort of the query, including the sort direction of individual fields. It is equivalent to the ORDER BY clause in SQL.
- The **Join** tab shows the join expressions used in the query. When you drag and drop joins between tables in the data model, the contents of this tab are defined for you. You can also express them manually.

Menu, Toolbar and Status Bar

The toolbar along the top of the Visual Query Builder (VQB) window provides clipboard icons, an icon to Run the query and a link to the help system. It also provides an Alias drop-down to select different aliases and a table drop-down to select tables from that alias to add to the query.

The VQB itself has no menu. But you can switch back to the SQL Editor window at any time for a menu of SQL-specific functions. Query properties are available in a dialog from this menu. These properties include:

- Static result table or live query view (if possible)
- Result table type
- Result table name and directory
- Remote query handling options
- Auxiliary tables for INSERT, UPDATE and DELETE queries
- Constrained updates for SQL views

Many areas of the VQB support Right-Click (RC) Menus to display additional options. For example:

- In the data model panel, RC on the table name/alias to remove the table or edit the alias.
- In the data model panel, RC on a join line to delete the join.
- In the Criteria panel, RC on a criteria line to drill down or delete the row, or to change the row type.
- In the Selection panel, RC on a selection line to delete the row or change the field type.
- In the Group Criteria panel, RC on a criteria line to delete the row or change the row type.
- In the Join panel, RC on a join row to delete the join expression.

The Corel® Paradox® 8 status bar also provides information about the VQB and the current task.

Errors in your query definition are dynamically shown in red within the various grids. This allows you to see an error immediately and correct it. (You should fix the problem or delete that defined element of the query immediately before proceeding to add new elements.)

Mapping the Elements of a SQL Statement

Here is a typical SQL Statement that has been annotated to show where each part of the statement is specified as part of the VQB.

```
SELECT Stock."Stock No", Stock.Description, SUM(
Lineitem.Qty ) Lineitem."1991 Qty"
```

Listed on the Selection panel of the tabbed notebook. Projected fields can be specified by clicking a check mark beside the field in the table representation in the data model. Summary fields and aliases are specified directly in the Selection panel.

```
FROM "STOCK.DB" Stock
```

```
INNER JOIN "LINEITEM.DB" Lineitem
ON (Stock."Stock No" = Lineitem."Stock No")
INNER JOIN "ORDERS.DB" Orders
ON (Lineitem."Order No" = Orders."Order No")
```

Tables are placed in the data model and joined by dragging from one table to the other. Joins are listed on the Join panel of the tabbed notebook. This is also where you can specify inner, left outer, right outer or full outer joins.

```
WHERE Orders."Sale Date" BETWEEN '01/01/1991'
AND '12/31/1991'
```

Selection criteria are specified on the Criteria panel of the tabbed notebook. You can drag fields from the data model to this grid, then specify the details of the expression.

```
ORDER BY Stock."Stock No", Stock.Description
```

Ordering of the results table is specified on the Sorting panel of the tabbed notebook. You can also drag fields from the data model to this grid. For each field in the sort expression, you can specify an ascending or descending sort.

```
GROUP BY Stock."Stock No", Stock.Description
```

GroupBy operations are specified on the Grouping panel of the tabbed notebook. When you specify an aggregate operator in the Selection panel, the non-aggregate fields are filled in on the Grouping panel for you.

```
HAVING SUM( Lineitem.Qty ) >= 20
```

Specified on the Group Criteria panel of the tabbed notebook. You must specify these criteria fields yourself, by typing or dragging the field from the data model.

Note that the name and type of the results table (:PRIV:ANSWER by default) are specified in the SQL Properties dialog, which can be reached from the SQL Editor by selecting SQL | Properties... from the menu.

Dialect Specific SQL

The Visual Query Builder (VQB) will create dialect-specific SQL, based on the back-end database being queried. For example, if you query Paradox tables, the SQL variant supports quoted aliases and fields with embedded spaces. If your query is performed against remote tables such as Oracle or SQL Server, the parser supports the SQL dialect specific to that server. In some instances, you will even see two versions of the SQL statement in the SQL Editor window, one that is used by the parser to update the VQB and one that is passed to the server, with slight but important differences between them.

The Project Viewer allows you to multiselect types in the left panel using CRTL-Click or simply by clicking and dragging. This is useful when you need to see all project files but you don't want to be bothered with non-Paradox file types or tables' auxiliary files. (*.MB, *.PX, *.TV, *.VAL, etc.)

The Project Viewer uses the Windows Explorer context menu to create file association. The right mouse button option for file association can be customized for Corel Paradox 8 file types. For example, to create and add a new option that will open QBE files in Notepad:

1. Open Windows Explorer.
2. Locate a QBE file.
3. Press shift and right click.
4. Choose the Open With option.
5. Select Notepad.

This change will be reflected in the Corel Paradox 8 Project Viewer.

Defining the Query's Data Model

The data model is specified in the top panel of the Visual Query Builder (VQB).

In defining the query's data model, the first step is to pick an existing alias from the **Database**: drop-down in the toolbar. You can also type a subdirectory name directly into this control. You can

even change the alias or directory while you are defining the query, in order to join tables from different databases.

The second step is to pick a table from the specified alias or directory. Click on the **Table**: drop-down to display a list of tables in that location. When you select a table, it is added to the query's data model.

Both drop-down edit controls also support incremental typing. As you start entering characters, they will show the first matching entry, allowing you to press [Enter] to select it.

Figure 3: A data model of three tables

Each table object displays the names and types for each field in the table. You can perform the following operations on a table object:

- Select individual fields by clicking on the field name.
- Select a range of fields by [Shift]-clicking.
- Select more than one field in a discontiguous list by [Ctrl]-clicking.
- Scroll the list to view other fields.
- Project all fields into the results table by clicking in the check box beside the table name.
  - The table check mark will display in blue.
- Project a single field into the results table by clicking in the check box beside the field.
  - The table check mark will display in light grey.
- Unproject fields by clicking on an already-checked box
- Minimize the table by clicking the UpArrow to the right of the table name.
- Remove the table by right-clicking and selecting Remove Table.
- Change the table alias by right-clicking and selecting Edit Table Alias, and typing a new name.
- Move the table objects around the data model pane by clicking and dragging on the name/alias.
- Expand or shrink the data model window by clicking and dragging one the of the edges.

Joining Tables

To join tables, click and drag the join field from one table object to the other. That join will be displayed with a line linking the tables objects, and the join fields will move to the top of the list and be bolded. If you have more than two tables, you can view each join in turn by clicking to select it.

To view the join expression, click on the Joins panel of the tabbed notebook. This grid displays the join fields for a single join at a time. To select a different join, click on the line in the data model, or click on the drop-down above the grid.

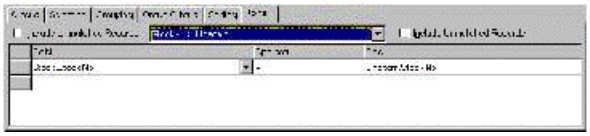


Figure 4: A query of three joined tables.

To specify outer joins, check one of the **Include Unmatched Records** boxes. Check the left box for a left outer join; the right box for a right outer join and both boxes for a full outer join. By default, neither box is checked and this provides an inner join.

Note that the Visual Query Builder (VQB) does some type checking, to stop you from joining fields which are not compatible. However, you can still join compatible fields that do not contain matched values.

To delete a join, you can do one of two things:

- Right click on the join line and select **Delete Join**
- Select the Joins panel of the tabbed notebook. Select the join in the grid, right click and select **Delete Row**

Self Joins

To specify a self-join, add the table to the data model twice, then join one object to the other on the field you wish to use.

For example, the following query joins Stock to itself to find those items that are more than a "Marine Magnetometer." This query requires a range join on the ListPrice field, as shown in Join panel of the tabbed notebook.

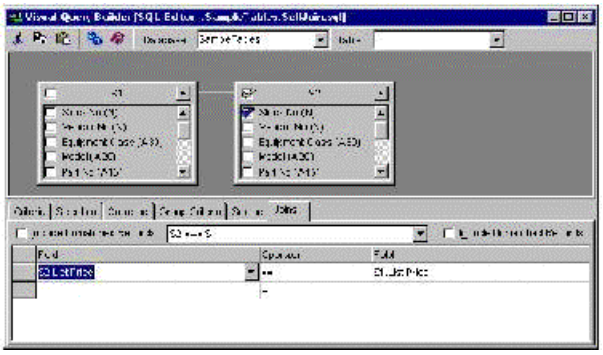


Figure 5: A self-join on the Stock table.

Here is the matching SQL statement:

```
SELECT S2."Stock No", S2.Description, S2."List Price"
FROM "STOCK.DB" S1
INNER JOIN "STOCK.DB" S2
ON (S2."List Price" >= S1."List Price")
WHERE S1.Description = 'Marine Magnetometer'
```

Another approach that may be easier for some users is to place the table once in the data model and then in the Joins panel of the tabbed notebook. Both drop-down lists contain fields from the table just placed. If you select the same field from each list a self-join is created. The Visual Query Builder (VQB) will add a second instance of the table to the data model for you, with the join line automatically placed.

Specifying Query Criteria

Use the Criteria panel of the tabbed notebook to specify query criteria. Three criteria types can be specified:

Simple Equation
Multiple Criteria
SQL Expression

Simple Equation

By default, the Criteria grid is configured for simple equations. On the left side, specify a field or literal. You can drag fields from the data model into this control, type the table and field name manually or click on the drop-down arrow to display a list of available fields. (If you type a field name directly, it is expanded to include the tablename, to avoid ambiguity.)

In the Compare column, specify a comparison operator from the following list:

- Equals (=) or Not Equals (<>)
- Range Comparison (>, <, >= or <=)
- LIKE or NOT LIKE
- IN or NOT IN
- BETWEEN or NOT BETWEEN
- IS NULL or IS NOT NULL

In the right column, specify a field or literal against which to perform the comparison. If one of the BETWEEN operators is selected, two columns allow you to specify the minimum and maximum values. If one of the IN operators is selected, specify a comma-separated list of values.

String criteria do not need to be quoted; the Visual Query Builder (VQB) will add quotes for you automatically. Dates must be quoted to distinguish them from algebraic expressions or strings.

Multiple Criteria

When you specify multiple criteria, you can designate how the query parser will apply all of the specified conditions. This is done from the drop-down list at the top of the Criteria grid.

- When **All** is selected (the default), the conditions are AND'ed (i.e. all must be met for the record to be selected.) Notice the AND keywords appearing in the Row Info column on the left side. This is equivalent to the following WHERE expression:

```
WHERE (Orders."Sale Date" < '06/01/1990')
AND (Orders."Ship VIA" = 'U.P.S.')
AND (Orders."Balance Due" < 10)
```

- When **Any** is selected, the conditions are OR'ed (i.e. any one or more than one of them can be met for the record to be selected.) Notice the OR keywords appearing in the Row Info column. This is equivalent to the following WHERE expression:

```
WHERE (Orders."Sale Date" < '06/01/1990')
OR (Orders."Ship VIA" = 'U.P.S.')
OR (Orders."Balance Due" < 10)
```

- When **None** is selected, the conditions are NOT OR'ed (i.e. all records except the ones where any conditions apply will be selected.) Notice the NOT keyword at the top of the grid and the OR keywords against other rows. This is equivalent to the following WHERE expression:

```
WHERE NOT( (Orders."Sale Date" < '06/01/1990')
    OR (Orders."Ship VIA" = 'U.P.S.')
    OR (Orders."Balance Due" < 10) )
```

- When **Not All** is selected, the conditions are NOT AND'ed (i.e. all records except the ones where all of the conditions apply will be selected.) Notice the NOT keyword at the top of the grid and the AND keywords against other rows. This is equivalent to the following WHERE expression:

```
WHERE NOT( (Orders."Sale Date" < '06/01/1990')
    AND (Orders."Ship VIA" = 'U.P.S.')
    AND (Orders."Balance Due" < 10) )
```

These options make it easier to express multiple grouped conditions without having to specify complex nesting or parentheses within the query.

SQL Expression

Use this option to type a WHERE clause or portion thereof directly into the SQL statement. Use this option if you are more comfortable typing in an expression than specifying it using the prompts, or if the expression is too complex to be easily expressed, or if your SQL back-end provides options that are not supported by the VQB.

EXISTS Clause

This option allows you to specify a SQL subquery that returns True or False. EXISTS is a comparison predicate that allows you to return a result set based on the existence of any value (but at least 1) from a subquery.

Criteria Grouping and Nesting

When you have two or more criteria, the VQB allows you to group them into a single expression, and then to drill down to view the details of that expression in the query builder.

To group two or more rows, [Shift]-click in the grey column to the left of the grid, so that both rows are selected. Right-click and select **Group Rows** from

the menu. The display will change as shown in the figure below, with a SQL expression defining the grouped criteria.

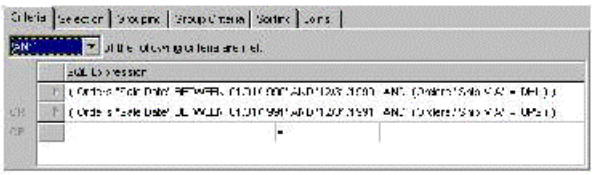


Figure 6: Grouped criteria into a SQL Expression

Notice the arrow in the left-most column. When you click on this arrow (or right-click and select **Drill Down** from the menu), the expression is expanded into separate criteria rows, in a child grid.

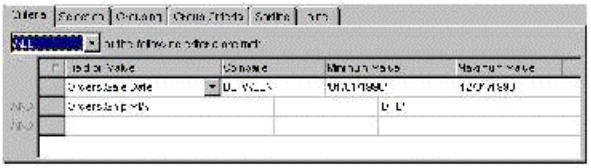


Figure 7: Expanded grouped criteria

Notice the back pointing arrow in the top-left corner of the grid. When you click on this arrow, the drill down is reversed and the grouped criteria are redisplayed as a SQL expression.

Selecting Fields to Display

The fields that will appear in the results table are listed in the Selection panel of the tabbed notebook. The Visual Query Builder (VQB) provides a number of ways to specify these fields.

- In the Data Model panel, click on the check box beside the table name to include all fields from that table. The check mark is displayed in blue. The Selection grid will have a row for each field in the table.
- In the Data Model panel, click on the check box beside a specific field to include it. This check mark is displayed in blue. If some fields from the table are not included, the table check mark is displayed in grey.
- In the Data Model panel, click and drag a field down to the blank row of the Selection grid. The field is checked in the Data Model's table object, and a new row added to the Selection grid.
- In the Selection grid, click on the Field column, then display the drop-down list of available fields. Select one.
- Type the field name yourself. If you leave out the tablename (Table.Field), it will be added for you automatically to avoid ambiguity. Field names with embedded spaces should be quoted.
- Type a literal value in the column. A new field is appended to the results table, and all rows have this value.

The Selection grid displays two different types of rows representing fields in the results table:

Fields

These are fields that are projected from the tables without modification. The Output Name column can be used to change the name of the result table column. It is equivalent to the AS operator in SQL.

Summary

These are aggregate or summary operations. To change a column from Field to Summary, right click and select the option from the menu. A new Summary column is inserted in the grid, with support for the following summary operations:

- Sum
- Sum Distinct
- Count
- Count Distinct
- Avg
- Avg Distinct
- Min

- Max

When you specify an aggregate operation, all non-aggregate fields are automatically added to the Grouping grid for you, to ensure that SQL's grouping requirements are satisfied.

You can also type the aggregate operation directly into a “Field” row, and the VQB will automatically reconfigure the row as a Summary. Note that you may need to adjust the output name since the default name will likely not pass muster. Figure 8 below shows a Selection grid with different field options.

Figure 8: A typical Selection grid showing different options.

Removing Duplicates

Paradox's old QBE used Check and CheckPlus to control whether duplicates were included or not. As you saw in the first part of this paper, this syntax was confusing for many users. SQL uses the DISTINCT operator to control the same option. When you perform a

```
SELECT <field list>
```

duplicates are retained. When you perform a

```
SELECT DISTINCT <field list>
```

duplicates are ignored in generating the results. To achieve this effect in the VQB, check the **Remove Duplicates** check box on the Selection tab. This setting applies to the whole query.

Calculating New Fields

To calculate new fields, drag the existing field down to a new row, then specify the remainder of the calculation. In the figure above, note the calculated field to display ListPrices as they currently exist, and how they would be if multiplied by 10%. Note that you will likely need to adjust the Output Name for such a field.

Field Order

The order of fields in the Selection grid determines the order of fields in the results table. If you need to change this order, click in the leftmost column of the grid (the grey one) and drag the row to a different position.

Grouping

The SQL standard requires that all non-aggregate fields must be listed in the GROUP BY clause, even if some fields do not contribute to the actual grouping of the query.

When you specify an aggregate field in the Selection grid, the VQB automatically adds all non-aggregate fields to the Grouped On list on the Grouping panel. It is generally not necessary to visit this panel at all!

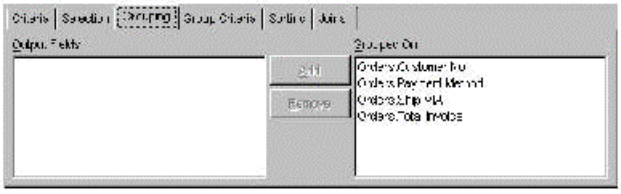


Figure 9: The Grouping panel.

Group Criteria

This panel allows you to specify a HAVING clause for the SQL query. It functions in the same way as the Criteria panel, except that each criterion is expressed as a summary operation. Figure 10 shows a typical Group Criteria panel, representing the following HAVING clause:

```
HAVNG SUM( LineItem.Qty ) > 20
```

Figure 10: The Group Criteria panel.

It is very rare to have more than one field represented in the HAVING clause and therefore in this grid. However, the VQB provides all of the criteria manipulation and grouping functions found in the Criteria panel.

Sorting

Paradox's old QBE made sorting very confusing. The overall sort order for the table was specified in a special dialog box, while the individual sort order of each field was specified by the Check and Check Descending or Calc and Calc Descending operators.

In SQL, sort order is specified via the ORDER BY clause. In the Visual Query Builder (VQB), the Sorting panel of the tabbed notebook allows you to control the sort order of the results table. When you switch to this panel, all fields slated to appear in the results table are listed in the Output Fields list on the left side.

To move a field from one list to the other, you can double-click or click on the Add and Remove buttons in between the two lists. The order of fields in the Sorted By list determines how the results table will be sorted.

For each field in the Sorted By list, you can also control the individual sort order, either ascending or descending. To change sort direction, highlight one or more entries in the Sorted By list and click on the a...z or z...a "buttons." You can also change the sequence of entries in this list by selecting an entry then clicking on the Up or Down arrows above the list.

Note that not every field in the results table needs to be moved to the Sorted By list. You should only move as many fields as are necessary to resolve sorting conflicts. (Even if you don't specify enough fields, Paradox will resolve remaining duplicates based on field sequence.)

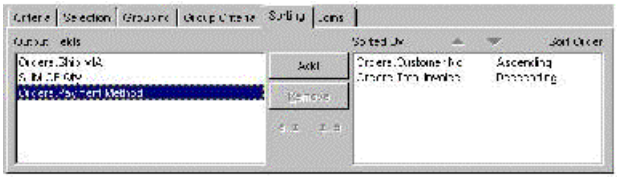


Figure 11: The Sorting panel.

Query Properties

Query properties are not specified in the Visual Query Builder (VQB). Instead, close the VQB and return to the SQL Editor. From there, select SQL | Properties from the menu (or click on the equivalent toolbar icon) to display the following dialog box:

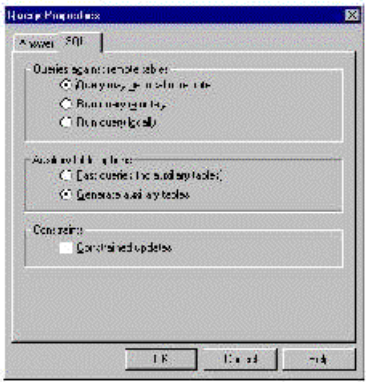


Figure 12: The Answer tab of the Query Properties dialog.

From this dialog, you can specify the following properties:

- Whether the result set is an Answer table (a copy of the queried records in a new table) or a live query view, which is a filter applied to the actual table so that changes affect the original records. There are strict rules as to which queries can be processed as live query views. For more information, see the online help.
- If the result set is an Answer table, you can specify the table type (Paradox or dBASE) and the name and subdirectory where this table will be placed. The default is ANSWER.DB, a Paradox table placed in your Private directory.

Viewing, Modifying, Saving and Loading the SQL

The Visual Query Builder (VQB) can only be invoked from the SQL Editor.

A Paradox SQL file is stored on disk as ASCII text. It has a commented header which includes the query properties and which is processed when the file is loaded in the SQL Editor. For example:

```
/*

Answer: :PRIV:ANSWER.DB
Type: PARADOX
Constrained: False
AuxTables: True
RunMode: Default
Alias: WORK
LiveAnswer: FALSE

*/

SELECT <etc...>
```

Each of the properties above corresponds to a setting in the SQL Query Properties dialog. Only settings which deviate from the defaults will normally be listed.

There are a number of different ways to access SQL files:

- From the Project Viewer, select SQL in the file types column. SQL files will be shown in the file list area. You can right click on a file to display a context-sensitive menu of options, including Run, Edit and Open (Edit and Run).
- If you right click on the SQL keyword in the Project Viewer, a popup menu with the New and Open options appears.
- You can also right click on the SQL Toolbar icon to display the same menu.
- Select **File | Open | SQL File**... from the menu.

The “New” options display a blank SQL Editor window. The “Open” options display the Open SQL File dialog, showing a list of SQL files in the current directory. When you select a file, it is opened into a new SQL Editor window. Once you have the SQL Editor window, click on the Visual Query Builder icon (or select the matching menu choice) to display the VQB.

When you have finished modifying your query, close the VQB. The final state of your query will be reflected in the SQL Editor, since Paradox keeps the two fully synchronized. From the editor window you can perform the expected functions:

- Save the file
- Save As to a different name
- Cancel changes
- Run the query
- Modify query properties
- Perform other functions (e.g. clipboard operations)

Subqueries

The Visual Query Builder (VQB) does not provide native GUI support for subqueries. However, subqueries can be defined if the nested query is typed directly. For example, consider the following

query that lists all stock items from the same vendor who supplies a specific item (# 1946):

```
SELECT S."Stock No", S."Vendor No", Model
FROM "STOCK.DB" S
WHERE S."Vendor No" =
( SELECT S1."Vendor No"
FROM "STOCK.DB" S1
WHERE S1."Stock No" = 1946 )
```

If you type this query into the editor, then load the VQB, the Stock table is added to the data model, the three output fields are added to the Selection grid and a single line is added to the Criteria grid, as shown in Figure 13:

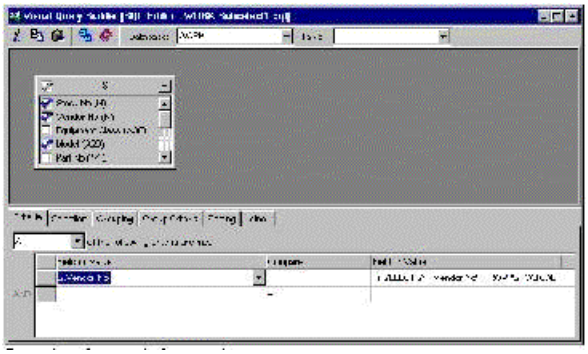


Figure 13: A subquery in the Criteria grid.

The complete sub-Select is specified in the Field or Value column of the Criteria grid.

Because you can specify different Compare operations in the Criteria grid, the VQB also supports range comparisons such as the following:

```
SELECT S."Stock No", Model, S."List Price"
FROM "STOCK.DB" S
WHERE (S."Equipment Class" = 'Small
Instruments')
AND (S."List Price" >
( SELECT AVG( S1."List Price" )
FROM "STOCK.DB" S1
WHERE S1."Equipment Class" = 'Small
Instruments' ))
```

Other compare operations that support subqueries include >=, <, <=, IN, NOT IN, BETWEEN, and NOT BETWEEN.

Correlated Subqueries

There is even limited support for correlated subqueries, as in the following example that finds all customers who have placed 12 or more orders:

```
SELECT C."Customer No", Name
FROM "Customer.DB" C
WHERE 12 <=
( SELECT COUNT( * )
FROM "Orders.DB" O
WHERE O."Customer No" = C."Customer No" )
```

Note however, that only the Customer table is added to the data model. The subquery is still expressed as a simple equation in the Criteria grid, with the SQL expression in the Field or Value column of the grid.